テスト環境の変化を考慮したソフトウェア信頼性モデルに関する一考察 (不確実性と意思決定の数理)

AUTHOR(S):

井上, 真二; 桑田, 裕文; 山田, 茂

CITATION:

井上, 真二 ...[et al]. テスト環境の変化を考慮したソフトウェア信頼性モデルに関する一考察 (不確実性と意思決定の数理). 数理解析研究所講究録 2009, 1636: 237-242

ISSUE DATE:

2009-04

URL:

http://hdl.handle.net/2433/140476

数理解析研究所講究録
第1636巻 2009年 237-242

テスト環境の変化を考慮したソフトウェア信頼性モデルに関する一考察

鳥取大学・大学院工学研究科　井上 真二 (Shinji Inoue)†
鳥取大学・大学院工学研究科　桑田 裕文 (Hirofumi Kuwada)†
鳥取大学・大学院工学研究科　山田　茂 (Shigeru Yamada)†
†Graduate School of Engineering, Tottori University

1 はじめに

ソフトウェア信頼度成長モデル（SRGM）[1–5] は，定量的なソフトウェア信頼性評価のための基盤技術の1つとして知られている．SRGM は，通常，テスト実行時間上におけるソフトウェア故障発生時間（もしくはソフトウェア故障発生時間間隔）を基本的な確率量として取り扱い，ソフトウェア故障発生時間の確率的性質がテスト期間中を通じて同一であるという仮定に基づいて構築される．しかしながら，開発管理面からの要因（納期遅れまたは納期までに目標とする信頼度が達成不可能と予測されるときなど）や固有技術面からの要因（フォールト発見難易度，フォールトの独立性，モジュール毎のフォールト密度の違いなど）によって，実際のソフトウェア開発のテスト工程では，ソフトウェア故障発生現象もしくはフォールト発見事象の統計的性質が著しく変化する現象がしばしば観測される．このような現象が発生するテスト時刻はチェンジポイント（change-point，以下 CP と略す）[6] と呼ばれ，これまでに提案されている SRGM に基づいた信頼性評価の精度に影響を与える要因の1つとして考えられている．

このような背景の下，近年では，上述した要因による CP 発生シナリオに基づいた SRGM [7–12] に関する議論がなされている．具体的には，CP 前後のソフトウェア故障発生率の違いのみを考慮した SRGM 構築枠組みに関する議論が多くなされている．しかしながら，実際のテスト工程では，テスト期間中においてテスト環境が変化しようともテストされているソフトウェア自体は同じであるため，CP 前後におけるソフトウェア故障発生時間には何らかの関係性があるものとしてモデリング枠組みを考える方がより現実的と言えよう．このような考え方は，基本的に，運用段階におけるソフトウェア信頼性評価手法 [13] の考え方とよく似ているものと考えられる，本研究では，文献 [13] のアプローチに基づきながら，CP 前後において異なるソフトウェア故障発生時間の関係性をある関数で与え，CP 前後におけるソフトウェア故障発生時間の関係性を陽にモデルへと反映できるような SRGM の構築枠組みについて議論する．最後に，実測データを用いながら提案モデルの適用例を示し，当該モデリング枠組みの有効性を簡単に議論する．

2 基本的な SRGM 構築枠組み

本研究で議論する CP を考慮した SRGM 構築枠組みは，基本的に以下の仮定 [14–17] に基づいて議論される．

(A1) ソフトウェア故障が発生した場合，その原因となるフォールトは，直ちにかつ完全に修正・除去される．

(A2) 各ソフトウェア故障は，それぞれ，独立かつ時間に関してランダムに発生して，各ソフトウェア故障発生時刻は，それぞれ，同一の確率分布 $F(t) \equiv \Pr\{T \leq t\} = \int_0^t f(x)dx$ に従う非負の確率変数によって与えられる．ここで，$f(t)$ は確率密度関数を表す．

(A3) テスト開始前にソフトウェア内に潜在する総フォールト数（初期潜在フォールト数）$N_0(> 0)$ は，ある確率分布に従う確率変数とする．

上記の3つの仮定は，テスト工程において検出可能なフォールトが有限の場合における SRGM 構築のための基本的仮定として知られている．

いま，$\{N(t), t \geq 0\}$ を任意のテスト時刻 t までに発見された総フォールト数を表す確率過程とする．このとき，(A1) ～ (A3) から，テスト時刻 t までに m 個のフォールトが発見される確率は，次のように定式

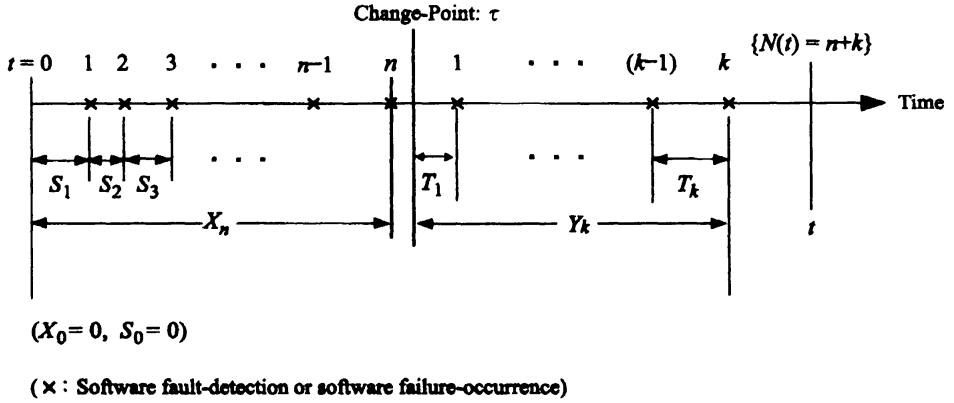

図 1： ソフトウェア故障発生現象およびフォールト発見事象に関する確率量.

化される.

$$\Pr\{N(t)=m\}=\sum_{n}\binom{n}{m}\{F(t)\}^{m}\{1-F(t)\}^{n-m}\Pr\{N_0=n\}\qquad(m=0,1,2,\cdots).\tag{1}$$

よく知られた結果として，(A3) において初期潜在フォールト数 N_0 が平均 $\omega(>0)$ のポアソン分布に従うと仮定した場合，式 (1) は，

$$\begin{aligned}\Pr\{N(t)=m\}&=\sum_{n}\binom{n}{m}\{F(t)\}^{m}\{1-F(t)\}^{n-m}\frac{\omega^n}{n!}\exp[-\omega]\\&=\exp[-\omega]\frac{\{\omega F(t)\}^m}{m!}\sum_{n}\frac{\{w(1-F(t))\}^{n-m}}{(n-m)!}\\&=\frac{\{\omega F(t)\}^m}{m!}\exp[-\omega F(t)]\qquad(m=0,1,2,\cdots)\end{aligned}\tag{2}$$

となり，平均値関数 $\mathrm{E}[N(t)]=\omega F(t)$ をもつ非同次ポアソン過程（nonhomogeneous Poisson process，以下 NHPP と略す）と本質的に等価となる．ここで，$\mathrm{E}[\cdot]$ は期待値を表す．したがって，当該モデリング枠組みに基づいて，ソフトウェア故障発生現象が NHPP に従う SRGM を構築したい場合は，式 (2) において，ある適切なソフトウェア故障発生時間分布 $F(t)$ を与えることによって NHPP モデルを構築することができる.

3 チェンジポイントを考慮した SRGM の構築枠組み

2 において議論した SRGM 構築のための基本的仮定 (A2) を拡張することで，CP を考慮した SRGM の構築枠組みを与える．まず，CP を考慮した場合のソフトウェア故障発生時間に関する確率量を次のように定義する.

X_i : CP 前における i 番目のソフトウェア故障発生時刻 $(X_0=0,\quad i=0,1,2,\cdots)$,

S_i : CP 前における i 番目のソフトウェア故障発生時間間隔 $(S_i=X_i-X_{i-1},\quad S_0=0,\quad i=1,2,\cdots)$,

Y_i : CP 後における i 番目のソフトウェア故障発生時刻 $(Y_0=0,\quad i=0,1,2,\cdots)$,

T_i : CP後におけるi番目のソフトウェア故障発生時間間隔 $(T_i=Y_i-Y_{i-1},\quad T_0=0,\quad i=1,2,\cdots)$.

図 1 に，テスト時間軸上において定義されたこれらの確率量をそれぞれ示している．ここで，τ は CP を表す．本研究では，上述した CP 前後の確率量が，それぞれ,

$$Y_i=\alpha(X_i),\qquad T_i=\alpha(S_i),\qquad J_i(t)=K_i(\alpha(t))\tag{3}$$

のような関係にあるものと仮定する．ここで，$\alpha(t)$ は CP 前後におけるソフトウェア故障発生時刻もしくはソフトウェア故障発生時間間隔の関係性を表すテスト環境関数，$J_i(t)$ および $K_i(t)$ はそれぞれ確率変数 S_i および T_i に対する確率分布関数を表す．

本研究では，テスト環境関数が $\alpha(t) = \alpha t(\alpha > 0)$ の場合 [13] を考える．ここで，α は比例定数であり，CP におけるテスト環境要因の変化が CP 前後のソフトウェア故障発生時間間隔の変化に与える影響の相対的大きさを表すパラメータである．いま，CP までに n 個のフォールトが発見され，それぞれのソフトウェア故障発生時刻が $0 < x_1 < x_2 < \cdots < x_n \leq \tau$ であるとする．このとき，CP から $n+1$ 番目のフォールトが発見される時間間隔 T_1 の確率分布関数は，

$$\begin{aligned}\bar{G}_1(t) \equiv \Pr\{T_1 > t\} &= \frac{\Pr\{S_{n+1} > \tau - x_n + t/\alpha\}}{\Pr\{S_{n+1} > \tau - x_n\}} \\ &= \frac{\exp[-\{M_B(\tau + t/\alpha) - M_B(x_n)\}]}{\exp[-M_B(\tau) - M_B(x_n)]}\end{aligned} \tag{4}$$

と求められる．ここで，$\bar{G}_1(t)$ は確率分布関数 $G(t) \equiv \Pr\{T_1 \leq t\}$ の余関数であり $\bar{G}_1(t) \equiv 1 - G_1(t)$ である．また，$M_B(t)(\equiv \omega F(t))$ は NHPP の平均値関数であり，CP 前において発見された総期待フォールト数を表す．式 (4) より，CP 後における任意のテスト時刻 $t \in (\tau, \infty]$ において発見される総期待フォールト数を表す NHPP の平均値関数は，

$$\begin{aligned}M_A(t) &= -\log \Pr\{T_1 > t - \tau\} \\ &= -\log \bar{G}_1(t - \tau) \\ &= M_B\left(\tau + \frac{t-\tau}{\alpha}\right) - M_B(\tau)\end{aligned} \tag{5}$$

と導出される．以上より，CP を考慮した NHPP の平均値関数は，

$$\Lambda(t) = \begin{cases}\Lambda_1(t) = M_B(t) & (0 \leq t \leq \tau) \\ \Lambda_2(t) = M_B(\tau) + M_A(t) = M_B\left(\tau + \frac{t-\tau}{\alpha}\right) & (t > \tau)\end{cases} \tag{6}$$

のようになる．また，指示関数を用いた場合，式 (6) は次のように表現される：

$$\Lambda(t) = \Lambda_1(t)U_1(x_1) + \Lambda_2(t)U_2(x_2). \tag{7}$$

ただし，$U_1(x_1)$ および $U_2(x_2)$ は指示関数を表し，それぞれ，

$$U_1(x) = \begin{cases}0 & (x < 0) \\ 1 & (x \geq 0),\end{cases} \qquad U_2(x) = \begin{cases}0 & (x \leq 0) \\ 1 & (x > 0)\end{cases} \tag{8}$$

である．式 (7) より，CP 前において観測されたデータに適したソフトウェア故障発生時間分布を与えることによって，CP を考慮した SRGM を構築することができる．

4 信頼性評価尺度

ソフトウェア信頼性評価尺度は，定量的な信頼性評価を行う上で有用な評価尺度として知られている．ここでは，代表的な以下の 3 つの信頼性評価尺度について簡単に議論する．まず，期待残存フォールト数は，任意のテスト時刻 t におけるソフトウェア内の期待残存フォールト数を表す尺度であり，確率過程 $\{N(t), t \leq 0\}$ が式 (2) の NHPP に従う場合，一般的に次のように導出される．

$$\begin{aligned}M(t) \equiv \mathrm{E}[\bar{N}(t)] &= \mathrm{E}[N(\infty) - N(t)] \\ &= \omega - \Lambda(t).\end{aligned} \tag{9}$$

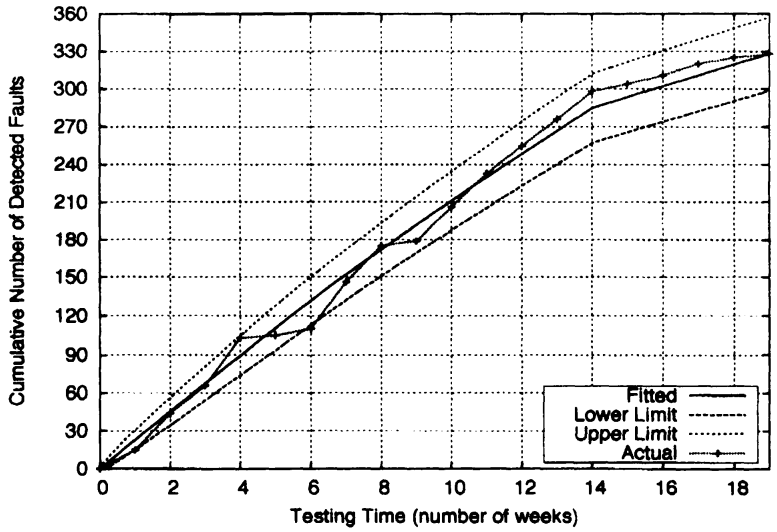

図 2： 推定された平均値関数とその 90% 信頼限界 ($\alpha = 2.0, \tau = 14$)．

ここで，$H(t)$ は NHPP の平均値関数を表す．また，ソフトウェア信頼度関数は，任意のテスト時刻 t までテストが進行しているという条件の下で，その後の時間区間 $(t, t+x](t \geq 0, x \geq 0)$ においてソフトウェア故障が発生しない条件付き確率と定義される．したがって，ソフトウェア信頼度関数 $R(x \mid t)$ は，

$$R(x \mid t) \equiv \sum_{k} \Pr\{N(t+x) = k \mid N(t) = k\} \Pr\{N(t) = k\}$$
$$= \exp[-\{\Lambda(t+x) - \Lambda(t)\}] \tag{10}$$

のように導出できる．最後に，累積 MTBF（mean time between software failures）は，ソフトウェア故障発生時間間隔分布が通常の確率分布関数の性質を満たさない場合における平均ソフトウェア故障発生時間間隔の代替的尺度の 1 つであり，

$$MTBF_C(t) = \frac{t}{\Lambda(t)} \tag{11}$$

として求められる．

5 適用例

実際のテスト工程において観測されたフォールト発見数データを用いて，今回提案したモデルに対する適用例を示す．本研究にて用いる実測データは 19 組のフォールト発見数データ $(t_k, y_k)(k = 0, 1, 2, \cdots, 19;\ t_{19} = 19$ (週)$, y_{19} = 328)$ [18] である．また，CP 前までにおけるソフトウェア故障発生時間分布がパラメータ b の指数分布に従う，すなわち，指数形 SRGM [19] に従う場合を考える．このとき，モデル内部に含まれるパラメータ ω および b の推定値 $\widehat{\omega}$ および $\widehat{b}$ は最尤法を用いて推定する．ただし，α および τ は所与としてそれらのパラメータを推定することにする．

本研究では，一例として，$\alpha = 2.0$ と仮定した場合の適用例を与える．図 2 に，上記の方法に基づいて推定された平均値関数とその 90% 信頼限界を示す．なお，CP は，実測データに対する平均偏差平方和 (MSE) が最も小さいときの値を適用している．図 2 から，発見フォールト数の実測データに沿って，CP 前後における発見フォールト数の平均的挙動は変化していることがわかる，また，図 3 に推定されたソフトウェア信頼度関数 $\widehat{R}(x \mid 19)$ を示す．図 3 より，当該ソフトウェアがテスト開始後 19 週目にリリースされ運用段階においてもテスト工程と同様の環境で使用した場合，リリース後 0.5 週目におけるソフトウェア信頼度は約 1.4815×10^{-2} と推定される．さらに，図 4 に推定された累積 MTBF を示す．図 4 より，テスト終了時刻における累積 MTBF は $\widehat{MTBF_C}(19) \approx 5.7927 \times 10^{-2}$ (weeks) と推定される．また，図 4 では，CP 以降，CP 以前よりも比較的急速にソフトウェア信頼度が向上していることが特に伺える．

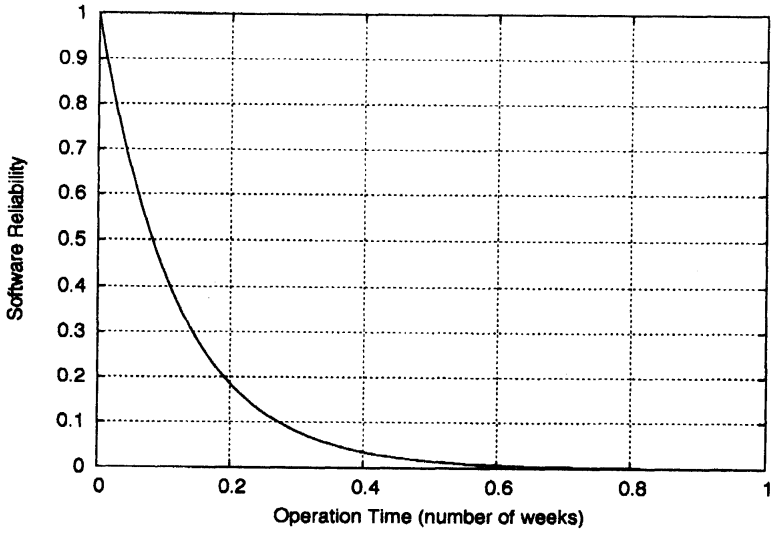

図 3： 推定されたソフトウェア信頼度関数 ($\alpha = 2.0, \tau = 14$)．

6 おわりに

本研究では，テスト環境の変化に起因して観測されるソフトウェア故障発生現象およびフォールト発見事象の確率的現象の著しい変化を考慮したソフトウェア信頼性モデルの構築を行った．具体的には，当該現象はソフトウェアのテスト環境と運用段階における使用環境との違いによく似ているため，文献 [13] において提案されたアプローチを適用した．当該アプローチを適用した利点としては，CP を考慮した従来のモデル [20] と比較して，CP 前後において異なるソフトウェア故障発生時間の関係性を陽にモデルへと反映できる点である．したがって，CP 前におけるソフトウェア故障発生現象を具体的に特徴付けると共に，CP 前後におけるソフトウェア故障発生時間の関係性を解析的に与えることによって，比較的容易にモデルを構築することができる．

今後は，実測データに基づきながら，CP 前後において異なるソフトウェア故障発生時間の関係性を表現するのに適切かつ具体的なテスト環境関数の構築を行う必要がある．また，今回議論したアプローチについて，CP を考慮したソフトウェア信頼性モデルとしての適合性および有用性についての検証を行う必要がある．

謝辞

本研究の一部は，日本学術振興会科学研究費補助金 若手研究 (B)（課題番号 19710129）および基盤研究 (C)（課題番号 18510124）の援助を受けたことを付記する．

参考文献

[1] J.D. Musa, D. Iannio, and K. Okumoto, *Software Reliability: Measurement, Prediction, Application.* McGraw-Hill, New York.

[2] S. Yamada and S. Osaki, "Software reliability growth modeling : Models and applications," *IEEE Trans. Soft. Eng.*, vol. SE-11, no. 12, pp. 1431–1437, 1985.

[3] 山田茂, ソフトウェア信頼性モデル —基礎と応用, 日科技連出版社, 東京, 1994.

[4] H. Pham, *Software Reliability.* Springer-Verlag, Singapore, 2000.

[5] 山田茂, 藤原隆次, ソフトウェアの信頼性：モデル, ツール, マネジメント, プロジェクトマネジメント学会 (PM 学会教育・出版シリーズ (1)) , 千葉, 2004.

[6] M. Zhao, "Change-point problems in software and hardware reliability," *Commun. Statist. — Theory Meth.*, vol. 22, no. 3, pp. 757–768, 1993.

[7] 大寺浩志, 山田茂, 成久洋之, "ソフトウェア信頼度成長モデルによるテスト工程管理," 電子情報通信学会論文誌, vol. J70-D, no. 5, pp. 889–895, 1987.

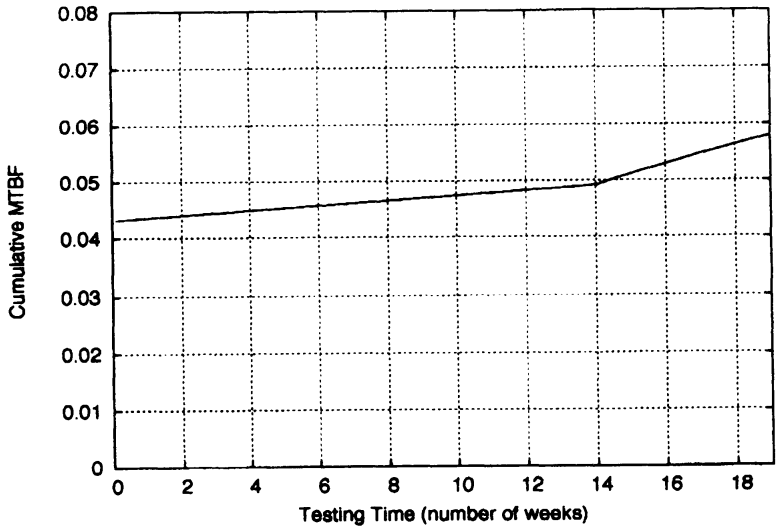

図 4： 推定された累積 MTBF ($\alpha = 2.0, \tau = 14$)．

[8] H. Ohtera and S. Yamada, "Optimal allocation & control problems for software-testing resources," *IEEE Trans. Reliab.*, vol. 39, no. 2, pp. 171–176, 1990.

[9] C.Y. Huang, "Performance analysis of software reliability growth models with testing-effort and change-point," *J. Sys. Soft.*, vol. 76, no. 2, pp. 181–194, 2005.

[10] C.Y. Huang, "Cost-reliability-optimal release policy for software reliability models incorporating improvements in testing efficiency," *J. Sys. Soft.*, vol. 77, no. 2, pp. 139–155, 2005.

[11] J. Zhao, H.W. Liu, G. Cui, and X.Z. Yang, "Software reliability growth model with change-point and environmental function," *J. Sys. Soft.*, vol. 79, no. 11, pp. 1578–1587, 2006.

[12] F.Z. Zou, "A change-point perspective on the software failure process," *Softw. Test., Verif. Reliab.*, vol. 13, no. 2, pp. 85–93, 2003.

[13] 岡村寛之, 土肥正, 尾崎俊治,「運用段階におけるソフトウェア製品の信頼性評価手法 —加速寿命試験モデルの提案—」, 電子情報通信学会論文誌, vol. J83-A, no. 3, pp. 294–301, 2000.

[14] N. Langberg and N.D. Singpurwalla, "A unification of some software reliability models," *SIAM J. Scien. Comput.*, vol. 6, no. 3, pp. 781–790, 1985.

[15] D.S. Miller, "Exponential order statistic models of software reliability growth," *IEEE Trans. Soft. Eng.*, vol. SE-12, no. 1, pp. 12–24, 1986.

[16] A.E. Raftery, "Inference and prediction for a general order statistic model with unknown population size," *J. ASA*, vol. 82, no. 400, pp. 1163–1168, 1987.

[17] H. Joe, "Statistical inference for general-order-statistics and nonhomogeneous-Poisson-process software reliability models," *IEEE Trans. Soft. Eng.*, vol. 15, no. 11, pp. 1485–1490, 1989.

[18] M. Ohba, "Software reliability analysis models," *IBM J. Res. Dev.*, vol. 28, no. 4, pp. 428–443, 1984.

[19] A.L. Goel and K. Okumoto, "Time-dependent error-detection rate model for software reliability and other performance measures," *IEEE Trans Reliab.*, vol. R-28, no. 3, pp. 206–211, 1979.

[20] S. Inoue and S. Yamada, "Software reliability measurement with change-point," *Proc. Fifth Intern. Conf. Qual. and Reliab.* (ICQR 2007), Chiang Mai, Thailand, November 5–7, 2007, pp. 170–175.

[21] S. Yamada, M. Ohba, and S. Osaki, "S-shaped reliability growth modeling for software error detection," *IEEE Trans. Reliab.*, vol. R-32, no. 5, pp. 475–478,485, 1983.